

# Airport Arrival Capacity Benefits Due to Improved Scheduling Accuracy

Larry A. Meyn\* and Heinz Erzberger†  
*NASA Ames Research Center, Moffett Field, CA, 94035*

**A new tool for studying scheduling logic and accuracy for terminal area arrival traffic is described. This tool, the Stochastic Terminal Area Scheduling Simulation (STASS), evaluated the benefits of improved scheduling accuracy for an arrival traffic rush period at the Dallas/Fort-Worth airport with current demand and several levels of increased demand. Four configurations of meter fix arrival accuracy and runway arrival accuracy were simulated: 1) a completely manual system, 2) a system utilizing Decision Support Tools (DST), 3) a highly automated system such as proposed in the Advanced Airspace Concept (AAC) and 4) an ideal system with perfect conformance of flights to scheduled positions. For delays equivalent to those produced by the manual system under today's demand, the DST, AAC and ideal configurations could accommodate increased demands of 19%, 42% and 69% respectively. The effect of accuracy and demand on the rate of potential runway violations is also presented. STASS was also used to study the benefits of scheduling flights so that some of their delay is adsorbed within Terminal Radar Approach Control (TRACON) airspace. The simulations show that absorbing some of the delay in the TRACON is beneficial as it increases runway utilization, which reduces overall delay and the average fuel burned.**

## I. Introduction

The long-term growth rate in air-traffic demand leads to future traffic densities that are unmanageable by today's air-traffic control (ATC) system. To accommodate such growth, new technology and operational methods will be needed in the next generation ATC system. One proposal for such a system is the Advanced Airspace Concept (AAC).<sup>1</sup> The AAC is designed to evaluate competing requests for access to the same airspace and resolve any conflicts while adhering to flow control constraints, including meeting scheduled times of arrival at specific waypoints or meter fixes. The trajectory information is fed directly into the aircraft's Flight Management System (FMS), which not only improves the speed and accuracy with which the information is entered but, it also allows greater flexibility in how such trajectories are defined. Previous studies have shown that this greatly improves the accuracy in meeting a scheduled time at a waypoint, reducing potential separation errors, enabling reduced spacing on final approach, all of which results in increased airport capacity.<sup>2-4</sup>

To estimate the potential increase in throughput due to the improved flight accuracies, a stochastic scheduling simulation was used. Previous studies of this type, such as the evaluation of scheduling algorithms for the Center-TRACON Automation System (CTAS), used the Fast-Time Simulation (FTS) code developed at the NASA Ames Research Center.<sup>5-7</sup> FTS was used to model various scheduling algorithms under a wide variety of conditions by randomly varying aircraft arrival times into the local airspace and then it stochastically modeled the accuracy of flights in meeting their scheduled arrival times at meter fixes. In this paper, a new tool, the Stochastic Terminal Area Scheduling Simulation (STASS), was used. The STASS is a re-implementation of FTS in the programming language Python. The purpose of the re-implementation was to allow more complex scheduling algorithms to be accurately modeled and to make it easier to change parameters defining the terminal area, making it easier to quickly model different airports.

The study presented here models the scheduling accuracy and performance of four systems designed to provide conflict free trajectories that will meet scheduled arrival times: 1) a completely manual system, 2) a system utilizing Decision Support Tools (DSTs) to assist the controller, 3) a highly automated system such as proposed in the AAC, and 4) an ideal system with perfect conformance of flights to scheduled positions. In addition to modeling the

---

\* Aerospace Engineer, AFM 210-10, AIAA Associate Fellow.

† Senior Scientist for Air Traffic Management, AFC 210-10, AIAA Fellow.

random errors in the times that flights arrive in a center and in their scheduled arrival time at meter fixes, the errors in meeting scheduled runway arrival times were also modeled. By modeling runway arrival time errors, the probability for violations of minimum separation standards under different scenarios can also be assessed.

## II. Simulation Description

### *Scheduling Overview*

For this study, STASS was configured to represent a scheduling algorithm of the type used by the Traffic Management Advisor (TMA) component of CTAS.<sup>7</sup> Fig. 1 shows the key features of the airspace represented by the scheduling model. The airspace is divided into two regions, Center airspace and Terminal Radar Approach Control (TRACON) airspace. Schedulers are implemented for both regions; these are referred to as the Center Scheduler and the TRACON Scheduler. The role of the TRACON Scheduler is to generate runway assignments and landing times to maximize runway utilization while allowing for safe separation between flights. The role of the Center Scheduler is to sequence flights into arrival streams at the meter fixes and generate meter fix arrival times. The Freeze Horizon shown in Fig. 1 can be defined as distance from the TRACON boundary, but it is usually defined as the location where a flight is a specific travel time from its assigned meter fix. Prior to a flight's arrival at the freeze horizon, the Center Scheduler can change its relative position in the sequence of flights assigned to a meter fix arrival stream. Once it is past the Freeze Horizon, its position in the meter fix arrival stream sequence is fixed.

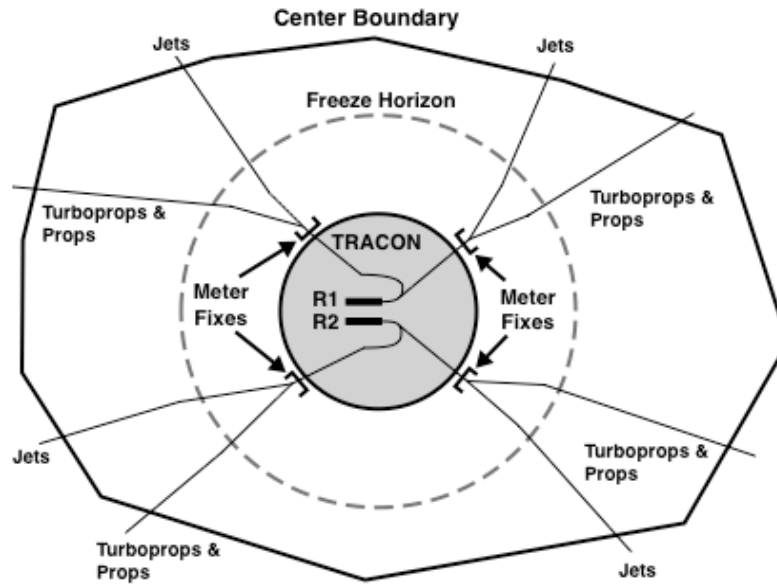


Figure 1 – Center/TRACON diagram.

The goals of the Center Scheduler are to maintain safe separation between flights, enable efficient runway utilization, and limit the amount of delay that flights may need to absorb while in the TRACON airspace. The last two goals of the Center Scheduler are conflicting. For runway efficiency, it's best to put as many flights as possible into a TRACON to increase the availability flights that can utilize available runway capacity. However, flights are at lower altitudes in TRACON airspace and use more fuel than when they are in the Center airspace, so if a flight needs to be delayed, it is less costly for it to absorb delay while in Center airspace. To balance these two goals, the Center Scheduler uses the Delay Distribution Function (DDF) described in Ref. 4. Basically, the DDF sets a maximum amount of delay,  $dT_{max}$ , that a flight can be scheduled to absorb within the TRACON. The Center Scheduler estimates the amount of delay that a flight needs to absorb to meet TRACON Scheduler constraints and sets meter fix arrival times so that any delay in excess of  $dT_{max}$  can be absorbed by the flight while it is in Center airspace. To accomplish this, the Center Scheduler generates a schedule that includes runway assignments; landing times and meter fix arrival times. This schedule needs to satisfy separation constraints at the runways and meter fixes, aircraft model-specific speed constraints and the DDF constraint that all delay up to  $dT_{max}$  is assigned to the TRACON portion of a flight's schedule.

If the schedule determined by the Center Scheduler could be followed precisely, there would be no need for a separate TRACON Scheduler and  $dT_{max}$  could be set to zero. However, for current and near-future ATC systems,

such precision is not feasible. Instead, flights arrive at meter fixes at their scheduled arrival time plus or minus some random error. To maintain efficiency, the TRACON Scheduler creates a new, optimized schedule based on when flights actually arrive at the meter fixes. Since flights may arrive at meter fixes later than scheduled, the TRACON Scheduler needs to have other flights available to replace them to maintain efficient runway use. When using the DDF, flights that are scheduled to absorb some delay in the TRACON may be available to replace late arriving flights in the landing sequence. The value of  $dT_{\max}$  is set to provide enough excess flights in the TRACON to maintain efficient runway utilization without having delayed flights spending excessive delay time in the TRACON where the fuel burn rate is higher.

### ***Scheduling Implementation and Inputs***

STASS models both Center and TRACON Schedulers, and it stochastically models the errors in flights meeting their scheduled arrival times at meter fixes and runways. The scheduling process has two parts. One is to take a sequence of aircraft with specific meter fix and runway assignments and determine the meter fix and runway arrival times that minimize delay and meet flight separation and speed constraints. For satisfying constraints, STASS utilizes the time constraint software described in the Appendix. This simplifies the expression of constraints and allows all constraints to be evaluated simultaneously when generating a schedule. The second part of the scheduling process is to try variations on flight sequences, meter fix assignments and runway assignments to see if a more efficient schedule can be generated. For this study, only runway assignments were varied. Meter fix assignments and re-sequencing flights were not included for reasons of expediency.

### **Constraints**

Four constraints are modeled: 1) separation between aircraft at runways, 2) TRACON transit times, 3) separation between aircraft at meter fixes and 4) Center transit times. Although separation constraints are commonly specified in terms of in-trail distance, these were transformed into corresponding in-trail separation times to allow all constraints to be represented as time constraints.

Runway separation minimums for four classes of aircraft are shown in Table 1. These are for aircraft landing in sequence on the same runway. For this study, the three runways modeled were considered independent and not subject to any inter-runway separation requirements. In addition to wake vortex separation minimums, a separation buffer can also be assigned. This buffer allows additional separation to be added to reduce the likelihood that separation minimums will be violated when flights miss their targeted runway arrival time. The expected arrival time at the runway is defined as the arrival time of the preceding flight plus the runway separation minimum plus the runway separation buffer.

Table 2 shows airport configuration data from Ref. 9, that was based on field trials of TMA at the Dallas/Fort Worth (DFW) airport in 1996. The data represent nominal TRACON transit times in seconds from each meter fix to each arrival runway. The data are different for jets and turboprops. If a runway is not available to a specific aircraft type at a meter fix, the time value is replaced by “---”. These values are used as minimum transit times in the schedulers. For maximum transit times,  $dT_{\max}$  is added to the minimum transit times. This constraint enforces the DDF by limiting TRACON transit times to be no more than  $dT_{\max}$  greater than the minimum transit time.

Meter fix separation minimums are derived from the FAA specified 5 nautical mile separation and a representative speed for aircraft for each stream class. For this study, the separation minimum was 72 seconds for jets, 85.7 seconds for turboprops and 100 seconds for piston aircraft. Separation buffers analogous to those defined for runway separations can also be specified; however, they were not used in this study. The minimum arrival time constraint at the meter fix is defined as the arrival time of the preceding flight in the same stream class plus the separation minimums for that stream class.

**Table 1 In-trail runway separations in seconds.**

Leader\Follower	SML	LRG	HVY
SML	58	54	50
LRG	84	54	50
HVY	140	108	72

**Table 2 TRACON transit times in seconds.**

<b>**EngineType: Jet</b>				
Runway\Fix	MF1	MF2	MF3	MF4
36L	649	723	592	---
35C	716	730	610	626
31R	820	645	---	621
<b>**EngineType: Turboprop</b>				
Runway\Fix	MF1	MF2	MF3	MF4
36L	715	812	587	---
35C	804	---	608	650
31R	---	723	---	625

The Center transit time constraint is dependent on the Freeze Horizon. In this study, the Freeze Horizon is defined as the point where a flight is nominally 19 minutes from its assigned meter fix. Therefore the minimum Center transit time constraint is a flight's time of arrival at the Freeze Horizon plus 19 minutes. No maximum Center transit time constraint is imposed, so no restriction is needed for the amount of time a flight can be delayed in the Center.

The Center Scheduler evaluates all four of these constraints based on the flight arrival times at the Freeze Horizon. The TRACON Scheduler evaluates the runway and TRACON transit time constraints based on actual meter fix arrival times. In the simulations presented, both the Center Scheduler and the TRACON Scheduler were free to assign flights to any available runway according to the information provided in Table 3. The selection criteria is to assign the runway that produces the earliest estimated landing time given the TRACON transit time constraints and the minimum in-trail separation constraints with respect to the immediately preceding flights on each runway. The TRACON Scheduler is free to change runway assignments based on meter fix Actual Time of Arrivals (ATAs) to optimize the landing sequence. The Center and TRACON Schedulers are not allowed to change the sequence of flights from a common meter fix arrival stream to a specific runway. This rule is set to avoid the situation where one flight overtakes another. However, flights from other meter fix arrival streams can be inserted into the landing sequence at a runway when a time slot is available. The assumption is that they are using a separate route to the runway final approach.

### **Stochastic Inputs**

The STASS code currently models three sources of stochastic variation. The first is random perturbations of the times at which flights arrive at the Freeze Horizon. The second is random variation of the accuracy of flights meeting their scheduled meter fix arrival times. The third is random variation of the accuracy of flights meeting their scheduled runway arrival times. Each stochastic simulation set consisted of 1000 simulation runs with random variations on individual flight arrival times at the Freeze Horizon, meter fixes and runways.

The nominal Freeze Horizon arrival schedule was based on the noon arrival rush at DFW used in Ref. 9. The variation of arrival times at the Freeze Horizon were based on a probability distribution generated by convolving three identical uniform distributions. This distribution approximates a Gaussian distribution, but only has a span of  $\pm 3$  standard deviations. The distribution generated for the variation in Freeze Horizon arrival times spanned  $\pm 240$  seconds and had a standard deviation of 80 seconds. To generate schedules for increased demand levels, the original schedule was randomly sampled to generate additional flights and produced a schedule with the same relative distribution of Freeze Horizon arrival times as the original schedule.

The stochastic variation in meter fix and runway arrival times used the same bell shaped distribution as was used for Freeze Horizon arrival times, but the standard deviations were chosen to represent the accuracies that can be achieved under four scenarios. The first scenario is for manual control with controllers giving voice communication of clearances. These clearances are based on the controllers' mental assessment of the traffic situation. The second scenario is for controllers using a DST that provides optimized schedules of meter fix arrival times, runway assignments and runway landing times, such as is provided by the Traffic Management Advisor (TMA).<sup>10</sup> The controllers then mentally determine strategies to meet the DST-provided arrival times and issue flight clearances, based on their strategies, via voice communication. The third scenario is for a highly automated system such as the proposed AAC system. In this system, fix and runway scheduling information are determined via automation, along trajectories designed to meet the schedule. These trajectories are then transmitted by data link to the aircraft. The fourth scenario is for perfect adherence to an automated schedule. These four scenarios are respectively referred to as Manual, DST, AAC and Ideal.

The estimated accuracies for these scenarios are presented in Table 3. The meter-fix arrival errors for Manual and DST accuracy estimates were derived from Ref. 2. The runway inter-arrival time errors for manual and DST accuracy estimates were derived from data presented in Ref. 11. The AAC accuracies were conservatively estimated as one half of the DST accuracies, although recent studies by Boeing<sup>12</sup> and others suggest that much smaller errors are achievable with FMS equipped aircraft. The runway arrival time errors are derived from the runway inter-arrival time errors by assuming the root sum square of two successive arrival time errors equals the inter-arrival time error.

**Table 3 Simulation accuracy parameters.**

Simulation Parameter	Manual Accuracy	DST Accuracy	AAC Accuracy	Ideal Accuracy
Meter fix Arrival Standard Dev. (sec)	±200	±100	±50	0
Runway Inter-arrival Time Standard Dev. (sec)	±20	±15	±7	0
Runway Arrival Time Standard Dev. (sec)	±13	±10	±5	0
Runway Separation Buffer (sec)	32	24	12	0

The runway inter-arrival time error is also used to determine the runway separation buffer. The basic premise is that the targeted inter-arrival separation should be greater than the minimum separation; otherwise, position errors would lead to a high occurrence of loss of minimum separation. This would happen 50% of the time if the error is distribution is symmetrical. This would be an unacceptable rate of separation violations because it would result in frequent interventions by pilots or controllers to maintain safe separation. However, if the probability of minimum separation loss is only 5%, then a special intervention would be required about once for every 20 landings. By assuming the inter-arrival time error is Gaussian, there is a direct relationship between the separation buffer and the probability of separation loss as shown in Fig. 2. This can be calculated using Student's t Distribution. For a 5% probability of separation loss, required separation buffer is estimated to be 1.645 times the standard deviation of the error. This is how the separation buffers in Table 3 were calculated.

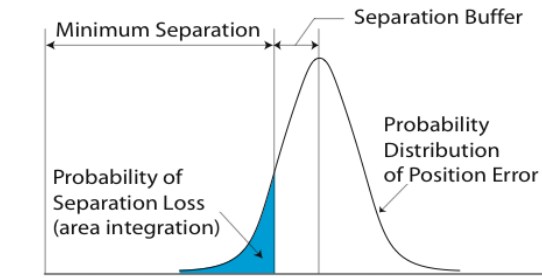


Figure 2 – Changing the probability of separation loss using a buffer.

Although this is a reasonable approach when separation is solely determined by in-trail horizontal separation, it should be noted that under Visual Flight Rules (VFR) the in-trail separation minimums are not applicable and not enforced by the FAA. Pilots are responsible for maintaining safe separation and are free to choose their own separation standards when issued a VFR approach clearance. So under VFR rules, the rationale for using a separation buffer is irrelevant. In a paper by Andrews and Robinson,<sup>13</sup> they estimated that the effective separation buffer to be a negative 7.1 seconds for the time period they studied, which was primarily under VFR conditions. For the study presented in this paper, visual separation by pilots was not considered.

### III. Results

A matrix of simulations included 4 sets of accuracy parameters as defined in Table 3 and a range of demands from the current-day demand to twice current-day demand. For each combination of accuracy and demand, 1000 simulations were run with random variation on freeze horizon, meter fix and runway arrival times. The analysis of each set of 1000 simulations included the generation of average demand, acceptance rate and delay curves such as the example shown in Fig 3. These curves were generated in the following manner: 1) for each simulation in a set, the number of unimpeded runway arrival times during a sliding 10 minute period was used to generate arrival demand, 2) the actual runway arrival times were used in similar fashion to generate the acceptance rate, 3) delay was determined from the difference between unimpeded and actual runway arrival times and these were averaged over a sliding 10 minute period, 4) the demand, acceptance rate and delay curves from each individual simulation in a stochastic set were averaged to produce the curves such as shown in Fig 3. The delay criterion used in subsequent analyses was the peak value of the average delay curve. This value was chosen as it represents an average of the worst delay a flight might see and, unlike an overall average, it should be independent of length of the period that was simulated. It should be noted that the acceptance rate is dependent on the sequence of aircraft types and the availability of flights to fill available landing time slots. This second factor explains why the acceptance rates increase with average delay. As delay increases, more aircraft are present to fill available landing time slots.

The average peak delay for manual, DST, AAC and ideal scheduling accuracies, as a function of demand, is presented in Fig 4. The demand is presented as a ratio to current day demand. The results are for  $dT_{max}$  of 2 minutes. The horizontal line in this figure is the average peak delay for manual scheduling accuracies. Based on the criterion

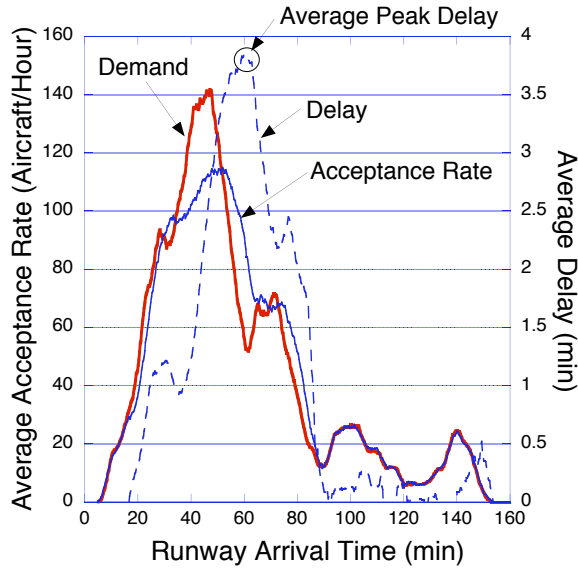


Figure 3 – Example of stochastically averaged demand, acceptance rate and delay curves for the noon arrival rush at DFW.

that average peak delay should be limited to that produced by manual accuracies based on today's demand, the increase in capacity enabled by DST, AAC and ideal scheduling accuracies are 19%, 42% and 69% respectively.

The rate of runway separation violations for the Manual, DST and AAC configurations as a function of demand is presented in Fig. 5. The first thing to note is that the violation rates are lower than the assumed 5% probability of separation loss that was used to determine the separation buffer. The primary reason is that there are periods in the schedule when flights are not always available to fill available landing slots. The prescribed 5% separation error rate would only occur if the airport was operating at its maximum capacity for the duration of the simulation. So the error rate is a function of the availability of flights for landing. Obviously, increasing demand can increase the availability, but increased delay also increases availability since flights are essentially waiting for a landing slot. It should be noted that AAC scenarios have consistently lower separation violations than the Manual and DST scenarios at all demand levels.

One of the reasons that the DDF was developed was to help minimize costs by reducing the amount of delay that flights adsorb in the TRACON where fuel burn rates are higher than in the Center airspace, yet to allow a sufficient amount of delay in the TRACON to increase the availability of flights to fill open slots in runway schedules. The amount of fuel burned while delayed is estimated using the following equation (Eqn. 34 in Ref. 4.)

$$F = 2d_C + 3d_T \quad (1)$$

Where  $F$  is the amount of fuel burned in lbs,  $d_C$  the seconds of delay adsorbed in the Center and  $d_T$  is the seconds of delay adsorbed in the TRACON. The average fuel burned for all aircraft while delayed as a function of  $dT_{\max}$  is presented in Fig. 6 for the four accuracy configurations with a demand ratio of 1.0. For all four accuracy configurations, the average fuel burned drops off with increasing  $dT_{\max}$  until a minimum is reached and then levels off. There are no clearly evident minimums where the fuel burned starts increasing with increasing  $dT_{\max}$ . This is different from the results presented in Ref. 4, but those results were for an airport operating with a single arrival

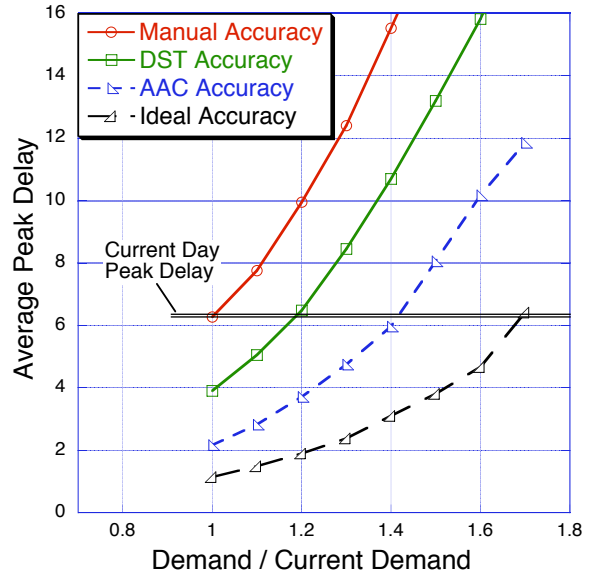


Figure 4 – Average peak delay as a function of demand,  $dT_{\max} = 2$  minutes.

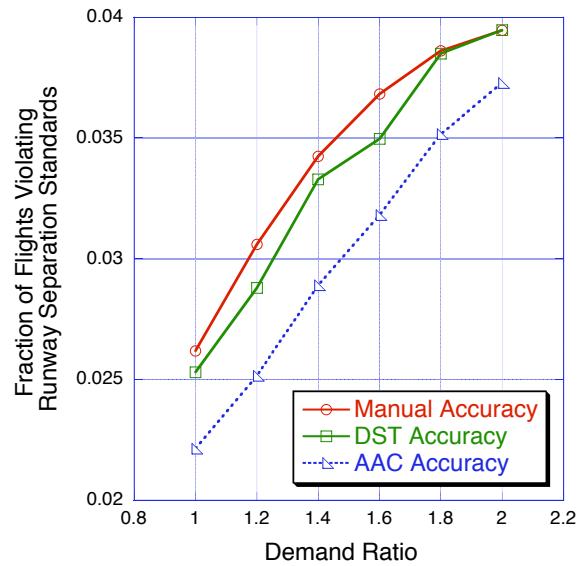


Figure 5 – Rate of minimum runway separation violations as a function of demand.

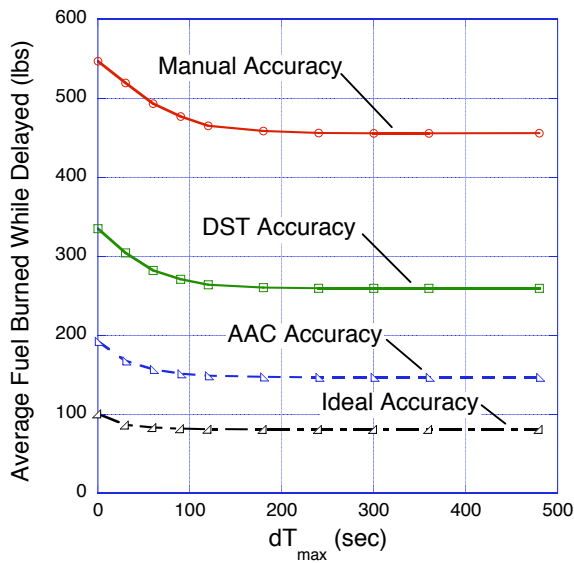


Figure 6 – Average fuel burned while delayed as a function of  $dT_{\max}$  for a demand ratio of 1.

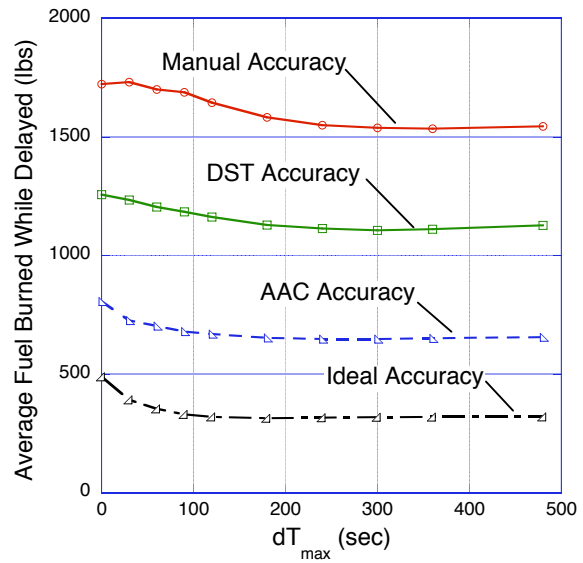


Figure 7 – Average fuel burned while delayed as a function of  $dT_{\max}$  for a demand ratio of 1.6.

runway. The ability to assign flights to alternative runways seems to increase the benefit of scheduling more delay in the TRACON and decrease the penalty of scheduling too much delay in the TRACON, a result also reported in Ref. 9. Even the Ideal accuracy configuration shows some benefit of using a  $dT_{\max}$  value of 90 seconds. This is most likely due to the increase in runway utilization that TRACON delay provides, which reduces overall delay.

The average fuel burned while delayed as a function of  $dT_{\max}$  is presented in Fig. 7 for the four accuracy configurations with a demand ratio of 1.6. Compared to Fig. 6, the average fuel burned is three to four times higher for the 60% increase in the demand. The curves for the Manual and DST configurations also seem to have more clearly defined minimum values. This seems to indicate that, when delays become excessive, even multiple runways systems reach a point where the increases in runway utilization provided by large values of  $dT_{\max}$  no longer compensate for the higher fuel burn rate in the TRACON.

#### IV. Conclusions

The STASS simulations indicate that, by improving arrival time accuracies, the AAC system could increase airport capacity by 42% with no increase over current day flight delays. This shows that schedule accuracy can have a significant impact on airport efficiency. The particular mix of accuracy parameters, airport configuration, fleet mix and flight schedule used in this study are not representative of all airports, or even all situations at a single airport, but the basic trends should apply. The trends are that improved scheduling accuracy can improve airport efficiency and can reduce potential separation violations. The trade off between improving efficiency and reducing the rate of potential separation violations was not explored. However, STASS would be an appropriate tool with which to perform such a study.

The role of scheduling some of a flight's expected delay for adsorption in the TRACON was also explored. In general, increasing  $dT_{\max}$  is beneficial up to a point, after which further increases provide no additional benefit. Further increasing  $dT_{\max}$  does not seem to increase average fuel burned, at least not for configurations with multiple arrival runways. However, at a demand ratio of 1.6, the Manual and DST accuracy configurations started to show increased fuel burn for large values of  $dT_{\max}$ .

STASS captures the essential effects that terminal area scheduling logic and errors in meeting scheduled times of arrival have on the efficiency of a terminal area. By only modeling the essential details of the scheduling, the simulation is fast enough to evaluate thousands of variations so that the stochastic behavior of different terminal area operational concepts can be explored. STASS has been shown to be a useful tool for the parametric analysis of the scheduling logic and scheduling accuracies for proposed air traffic management concepts.

#### References

1. Erzberger, H., "Transforming the NAS: The Next Generation Air Traffic Control System," 24th International Congress of the Aeronautical Sciences (ICAS 2004), Yokohama, Japan, Aug. 29 - Sept. 3, 2004.
2. Swenson, H. N., Hoang T., Engelland S., Vincent D., Sanders T., Sanford B., and Heere K., "Design and Operational Evaluation of the Traffic Management Advisor at the Fort Worth Air Route Traffic Control Center," 1st USA/Europe Air Traffic Management R&D Seminar, Saclay, France, June, 1997.

3. Davis, T.J., Krzeczowski K.J., and Bergh C., "The Final Approach Spacing Tool," IFAC Thirteenth Symposium on Automatic Control in Aerospace, Palo Alto, CA, September, 1994.
4. Erzberger, H., "Design Principles and Algorithms for Automated Air Traffic Management," AGARD Lecture Series 200 Presentation, Madrid, Spain, Paris, France, and Moffett Field, California, USA, November 1995.
5. Neuman, F., and Erzberger, H., "Analysis of Sequencing and Scheduling Methods for Arrival Traffic," NASA Technical Memorandum 102795, Ames Research Center, 1990.
6. Neuman, F., and Erzberger, H., "Analysis of Delay Reducing and Fuel Saving Sequencing and Spacing Algorithms for Arrival Traffic," NASA Technical Memorandum 103880, 1991.
7. Carr, G. C., Erzberger, H., and Neuman, F., "Delay Exchanges in Arrival Sequencing and Scheduling," *Journal of Aircraft*, Vol. 36, No. 5, pages 785-791, September-October, 1999.
8. Erzberger, H., Davis, T.J., and Green S.M., "Design of Center-TRACON Automation System," AGARD Meeting on Machine Intelligence in Air Traffic Management, Berlin, Germany, May 11-14, 1993.
9. Neuman, F., Erzberger, H., and Meyn, L., "A Fast-Time Simulation Tool for Analysis of Airport Arrival Traffic," NASA/TP-2004-212283, August 2004.
10. Wong, G. L., "The Dynamic Planner: The Sequencer, Scheduler, and Runway Allocator for Air Traffic Control Automation," NASA/TM-2000-209586, April 2000.
11. Credeur, L., Capron, W. R., Lohr, G.W., Crawford, D.J., Tang, D.A. and Rodgers Jr., W.G., "Final-Approach Spacing Aids (FASA) Evaluation for Terminal-Area, Time-Based Air Traffic Control," NASA TP-3399, December 1993.
12. Hughes, D., "Boeing Outlines Its Concept for Tripling ATC Capacity by 2025," *Aviation Week & Space Technology*, November 11, 2004, page 44.
13. Andrews, J.W., and Robinson III, J.E., "Radar-Based Analysis of the Efficiency of Runway Use," AIAA Guidance, Navigation & Control Conference, Montreal, Quebec, August 6-9, 2001.

## Appendix

### Time Constraint Modeling

The Stochastic Terminal Area Scheduling Simulation (STASS) tool utilizes a new, object-oriented representation of time constraints. The reason for this approach is that the procedural coding of interacting time constraints can be extremely tedious and prone to error, especially when more than just a couple of constraints are being modeled. In this situation an object-oriented approach can simplify the expression of complex data manipulations by encapsulating the methods needed to manipulate the data within the data objects. These manipulations then occur automatically whenever the data objects are used in calculations.

#### Time Constraint Objects

The time constraint module in STASS represents time constraints using two types of data objects, constraint pairs and constraint lists. A constraint pair is simply two values, `min` and `max`, which describe a valid time range for a constraint. The `min` value is the time that represents the minimum or earliest time that is valid, and `max` is the maximum or latest time that is valid. If the `min` value is "None," then there is no minimum constraint. If `max` is "None," then there is no maximum constraint. A constraint list is a set of constraint pairs representing all valid time ranges. For example, the set of available appointments times at a doctor's office might be from 1:00 PM to 1:30 PM and from 3:00 PM to 4:00PM. This would be represented as [(1:00 PM,1:30 PM), (3:00 PM,4:00 PM)].

Table A-1 lists five operations that can be used to combine two constraint pairs or to combine a constraint pair with a scalar value. The first operation listed is the logical `and` operation, in which two constraint pairs are combined to produce a constraint pair that satisfies both of the constraints being combined. For example, if two people need to find a time when they both can meet and the first is available from 1:00 PM to 3:00 PM and the second is available from 2:00 PM to 4:00 PM, the problem can be represented as the `and` operation for the constraint pairs (1:00 PM,3:00 PM) and (2:00 PM,4:00 PM), which produces (2:00 PM,3:00 PM). If there is no valid time range that satisfies both constraints, then a value of `None` is produced. The next two operations listed are for adding or subtracting a scalar value to a constraint pair shifts the valid time range by the scalar value. Finally, adding or subtracting a constraint pair to a constraint pair both shifts and expands the valid time range. The second operand can be thought of as a time range that is added or subtracted from a time constraint. For example, if a car is constrained to leave Point A from 1:00 PM to 1:15 PM and the travel time to Point B is constrained to be from 20 to 30 minutes then the arrival time at Point B will be (1:00 PM,1:15 PM) + (20,30). Using the operation for `add` in Table A-1 produces a Point B arrival time constraint of (1:20 PM, 1:45 PM). To get the earliest time of arrival, the car leaves at 1:00 PM and takes 20 minutes travel time to arrive at 1:20 PM. To get the latest time of arrival, the car leaves at 1:15 PM and takes 30 minutes travel time to arrive at 1:45 PM.

**Table A-1 Constraint pair operations.**

Operation Type	Implementation
<b>and ( &amp; )</b> with a constraint pair	$(\min_1, \max_1) \& (\min_2, \max_2) = (\text{maximum}(\min_1, \min_2), \text{minimum}(\max_1, \max_2))$
<b>add ( + )</b> with a scalar value	$(\min_1, \max_1) + x = ((\min_1 + x), (\max_1 + x))$
<b>subtract ( - )</b> with a scalar value	$(\min_1, \max_1) - x = ((\min_1 - x), (\max_1 - x))$
<b>add ( + )</b> with a constraint pair	$(\min_1, \max_1) + (\min_2, \max_2) = ((\min_1 + \min_2), (\max_1 + \max_2))$
<b>subtract ( - )</b> with a constraint pair	$(\min_1, \max_1) - (\min_2, \max_2) = ((\min_1 - \max_2), (\max_1 - \min_2))$

Table A-2 lists six operations that can be used to combine two constraint lists or to combine a constraint list with a scalar value. The six operations supported are the logical **and** and **or** operations in which two constraint lists are combined to produce a constraint list. The first operation listed is the **and** operation which produces the list of constraint pairs that represents the intersection of the constraint lists being combined. If there are no valid time ranges in the intersection of the two lists, then a value of **None** is produced. The second operation listed is the **or** operation which produces the list of constraint pairs that represents the union of the constraint lists being joined. The next two operations listed are for adding or subtracting a scalar value to a constraint list results in the scalar value being added or subtracted to every constraint pair in the list. Finally, adding (or subtracting) a constraint list to another constraint list produce the union of the addition (or subtraction) of each constraint pair in the first list to each constraint pair in the second list. When joining all the constraint pairs to produce the resultant list, overlapping constraint pairs are combined. For example, (1:00 PM, 3:00 PM) and (2:30 PM, 4:00 PM) would be combined to produce (1:00 PM, 4:00 PM).

**Table A-2 Constraint list operations.**

Operation Type	Implementation
<b>or (   )</b> with a constraint list	$\text{list}_1   \text{list}_2 = \text{join}(\text{list}_1, \text{list}_2)$
<b>and ( &amp; )</b> with a constraint list	$\text{list}_1 \& \text{list}_2 = \text{join}(\text{list}_1[i] \& \text{list}_2[j])$ for each <i>i</i> th pair in $\text{list}_1$ and <i>j</i> th pair in $\text{list}_2$
<b>add ( + )</b> with a scalar value	$\text{list}_1 + x = \text{join}(\text{list}_1[i] + x)$ for each <i>i</i> th pair in $\text{list}_1$
<b>subtract ( - )</b> with a scalar value	$\text{list}_1 - x = \text{join}(\text{list}_1[i] - x)$ for each <i>i</i> th pair in $\text{list}_1$
<b>add ( + )</b> with a constraint list	$\text{list}_1 + \text{list}_2 = \text{join}(\text{list}_1[i] + \text{list}_2[j])$ for each <i>i</i> th pair in $\text{list}_1$ and <i>j</i> th pair in $\text{list}_2$
<b>subtract ( - )</b> with a constraint list	$\text{list}_1 - \text{list}_2 = \text{join}(\text{list}_1[i] - \text{list}_2[j])$ for each <i>i</i> th pair in $\text{list}_1$ and <i>j</i> th pair in $\text{list}_2$

For simplicity, constraint list objects only use constraint pairs internally. Users only need to specify and use constraint lists. Python, like C++, supports operator overloading, so time constraint expressions can be written using the operators “&”, “|”, “+” and “-”. This helps make complex time constraint expressions very easy to write and understand.

### Application to Terminal Area Calculations

As an example, a terminal-area scheduling problem is presented. The constraints will include constraints at the meter fix, TRACON transit time constraints and constraints at the runway. All time values in this example will be expressed in seconds from a reference time.

At the meter fix there are two constraints, the earliest time that a flight can arrive and a minimum in-trail time separation from any previous aircraft crossing the meter fix. For this example, the earliest meter fix arrival time is 500 sec, the minimum in-trail time separation at the meter fix is 60 sec and the previous aircraft crossed the meter

fix at 480 sec. An interactive Python session for setting the meter fix constraint is shown in Listing A-1. For reference, line numbers are in the gray area preceding the text. When lines are too long to fit the display, they are continued on subsequent unnumbered display lines preceded by an ellipsis. User inputs are preceded by the “>>>” prompt, while outputs have no prompt. Lines 1 to 3 set the arrival time constraint, the in-trail separation constraint and the time that the previous aircraft crossed the meter fix. These are combined into a single constraint on line 4. The in-trail separation constraint and the previous aircraft meter fix crossing time are combined using an addition operator, “+,” the resultant constraint and the arrival constraint are combined using the logical and, “&,” operator. Line 5 is the command to print the resultant meter fix constraint that is shown on line 6.

```

1 >>> meterFixArrival = TimeRange(500, None)
2 >>> meterFixSeparation = TimeRange(60, None)
3 >>> previousMeterFixArrival = 480
4 >>> meterFixConstraint = arrivalConstraint &
... (previousMeterFixArrival + meterFixSeparation)
5 >>> print meterFixConstraint
6 [(540, None)]

```

Listing A-1, Listing of a Python session setting the meter fix arrival constraints.

Listing A-2 shows the session listing for setting TRACON transit time constraint. The minimum time to transit the TRACON is 400 sec, which is set in line 7. Up to 120 sec of delay can be added to the TRACON transit time. This delay constraint is set in line 8. These constraints are combined in line 9 and the result printed on line 11. The resulting constraint shows the minimum TRACON transit time is 400 sec, while the maximum is 520 sec.

```

7 >>> minimumTraconTransitTime = 400
8 >>> allowedTraconTransitDelay = TimeRange(0, 120)
9 >>> traconTransitTimeConstraint = minimumTraconTransitTime +
... allowedTraconTransitDelay
10 >>> print traconTransitTimeConstraint
11 [(400, 520)]

```

Listing A-2, Listing of a Python session setting TRACON transit time constraints.

To demonstrate the flexibility of time constraint object software, a more complex set of constraints are set for the runway. Rossow, et al.\* have proposed that future airport operations may include formation landings on closely-spaced parallel runways. Due to the hazard posed by the vortex wake shed of the leading aircraft, there are two safe regions for the following aircraft as depicted in Fig. A-1. If the in-trail separation of the following aircraft is short enough, the wake from the lead aircraft will not have spread far enough to impact the airspace of the follower. However, if in-trail separation is too long, then wake hazard region will impact the follower’s airspace and it will have to wait for a specified time period to allow the wake hazard to dissipate.

For this example there is a previously scheduled aircraft landing on the same runway as the aircraft being scheduled and a previously scheduled aircraft landing on a closely-spaced parallel runway. The session listing for entering these constraints is shown in Listing A-3. The previously scheduled aircraft on the same runway has a landing time of 700 sec, which is entered in line 12. The minimum separation constraint is for 180 sec, which is entered as a time constraint in line 13. The addition of these two values is done in line 14 and the resulting constraint is displayed in line 16. The previously scheduled aircraft on the adjacent runway has a landing time of 925 sec, which is entered in line 17. The formation flight separation constraint is from 0 to 30 sec after the leading aircraft, which is entered as a time constraint in line 18. If this time window cannot be made the constraint is that the minimum separation must be at least 180 sec. This is entered in line 19. The constraint for the adjacent runway is the landing time of the previous aircraft plus either the formation separation constraint or the adjacent runway constraint, which is expressed in line 20 and the result is displayed in line 22. This result shows two valid time constraints, the first for a formation landing and the second for a standard landing. The same runway constraint and the adjacent runway constraint are combined using a logical and in line 23, with the result displayed in line 25.

---

\* Rossow, V.J., Hardy, G.H. and Meyn, L.A., “Models of Wake-Vortex Spreading Mechanisms and Their Estimated Uncertainties,” AIAA-2005-7353, AIAA 5th Aviation, Technology, Integration, and Operations Conference, September 2005.

Finally, the meter fix, TRACON transit time and runways constraints are combined and evaluated in the listing presented in Listing A-4. Line 26 gives the expression for calculating the complete constraint at the meter fix. The TRACON transit time constraint is subtracted from the runway constraint and the result is combined with the meter fix constraint using a logical and. The result is displayed in line 28. In a similar fashion, the complete constraint at the runway is calculated in line 29 and the result displayed in line 31.

This example has shown how complex time constraints can be expressed using simple addition, subtraction and logical operations. This method of representing constraints should help reduce constraint programming errors and allow large numbers of interacting constraints to be easily expressed and evaluated.

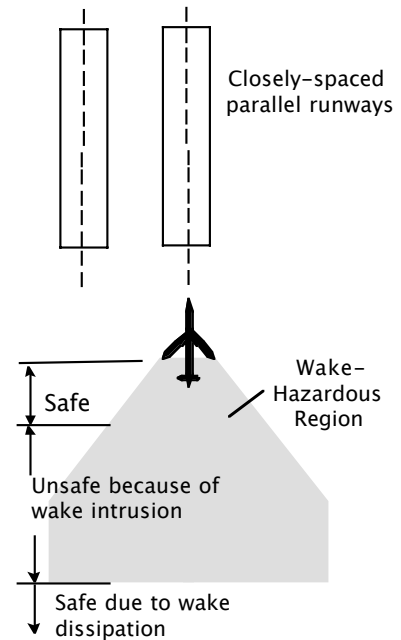


Fig. A-1, Safe and unsafe regions behind an aircraft landing on the adjacent runway of a closely-spaced runway pair.

```

12 >>> previousAircraftSameRunway = 700
13 >>> sameRunwaySeparation = TimeRange(180, None)
14 >>> sameRunwayConstraint = previousAircraftSameRunway +
...   sameRunwaySeparation
15 >>> print sameRunwayConstraint
16 [(880, None)]

17 >>> previousAircraftAdjacentRunway = 925
18 >>> adjacentRunwayFormationConstraint = TimeRange(0, 30)
19 >>> adjacentRunwaySeparationConstraint = TimeRange(180, None)
20 >>> adjacentRunwayConstraint = previousAircraftAdjacentRunway
...   + (adjacentRunwayFormationConstraint | adjacentRunwaySeparationConstraint)
21 >>> print adjacentRunwayConstraint
22 [(925, 955), (1105, None)]

23 >>> runwayConstraint = sameRunwayConstraint & adjacentRunwayConstraint
24 >>> print runwayConstraint
25 [(925, 955), (1105, None)]

```

Listing A-3, Listing of a Python session setting runway constraints.

```

26 >>> completeConstraintAtMeterFix = meterFixConstraint &
...   (runwayConstraint - traconTransitTimeConstraint)
27 >>> print completeConstraintAtMeterFix
28 [(540, 555), (585, None)]

29 >>> completeConstraintAtRunway = (meterFixConstraint +
...   traconTransitTimeConstraint) & runwayConstraint
30 >>> print completeConstraintAtRunway
31 [(940, 955), (1105, None)]

```

Listing A-4, Listing of a Python session for evaluations of the complete constraint set.